

# Effective Platform Engineering In Action

A practical journey through the essential components of a modern engineering platform, culminating in real-world automation and self-service capabilities.



# **Recap - Video # 1: Platform Engineering Fundamentals**

- Challenges and past trends that led to today's Platform Engineering
- Evolving PE in organizations
- Platform Engineering: Career Paths and Growth Opportunities
- Making it all stick
- The current and future trends in Platform Engineering
- How this book will help you?

# **Recap - Video # 2: Technical Platform Product Management**

- Why Platform Engineering Needs Product Management
- Driving business outcomes with product practices
- The platform value model
- Scaling engineering platform adoption

## **Recap - Video # 3: Compliance at the Point of Change**

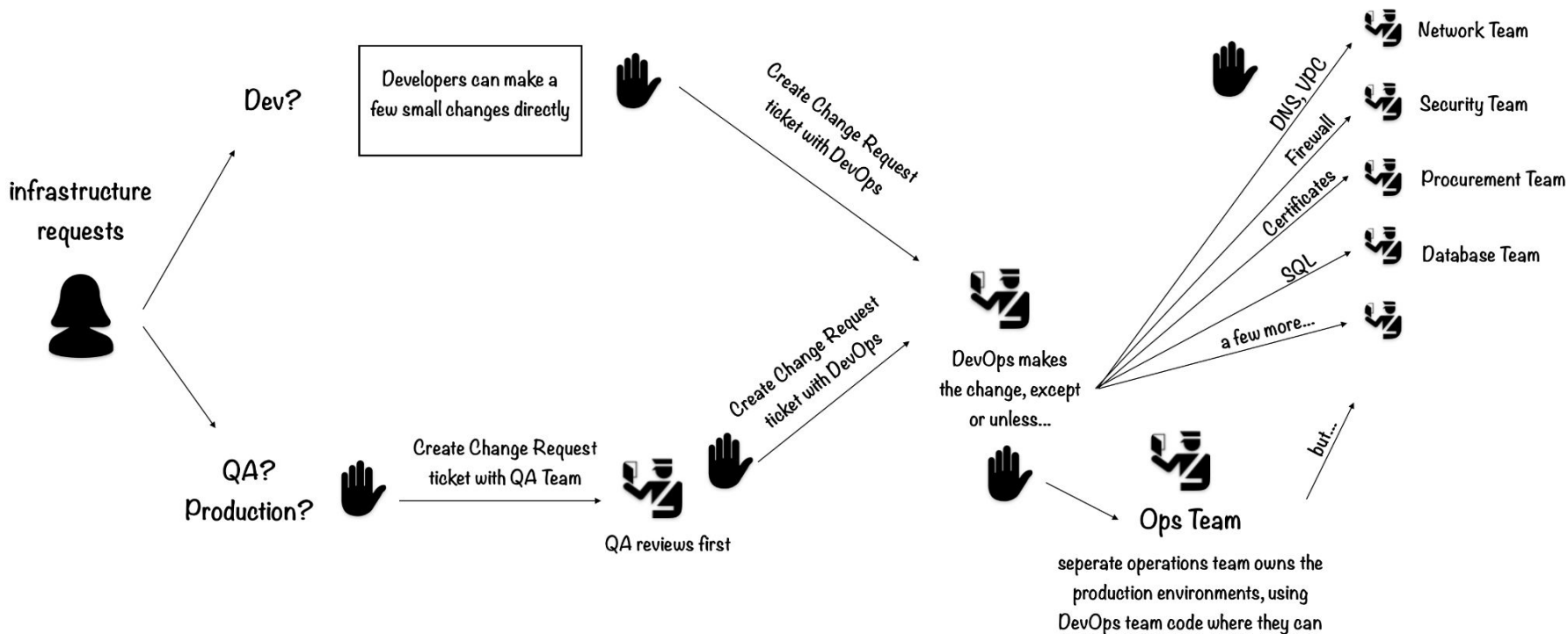
- Introduce compliance early in the SDLC
- Enforce policies using automation to detect and resolve issues before deployment
- Embed checks into dev tools and CI/CD pipelines to prevent non-compliance proactively and improve developer experience
- Compliance validation should be continuous, not static gates

# What We'll Cover

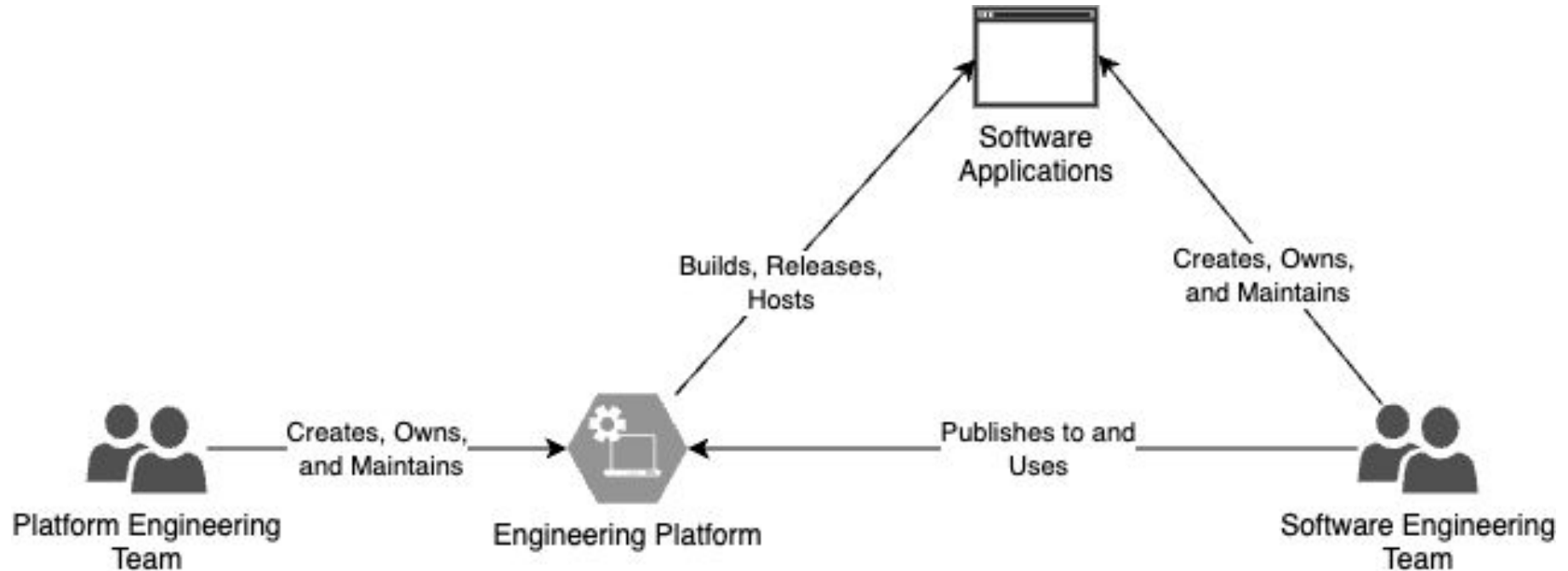
- ❖ Goals of an Engineering Platform
- ❖ Development practices for the platform
- ❖ Embedding observability
- ❖ Developing and deploying the control plane
- ❖ Components to extend the engineering platform for self-service

**What are the goals of an Engineering Platform?**

# What do we want to fix?

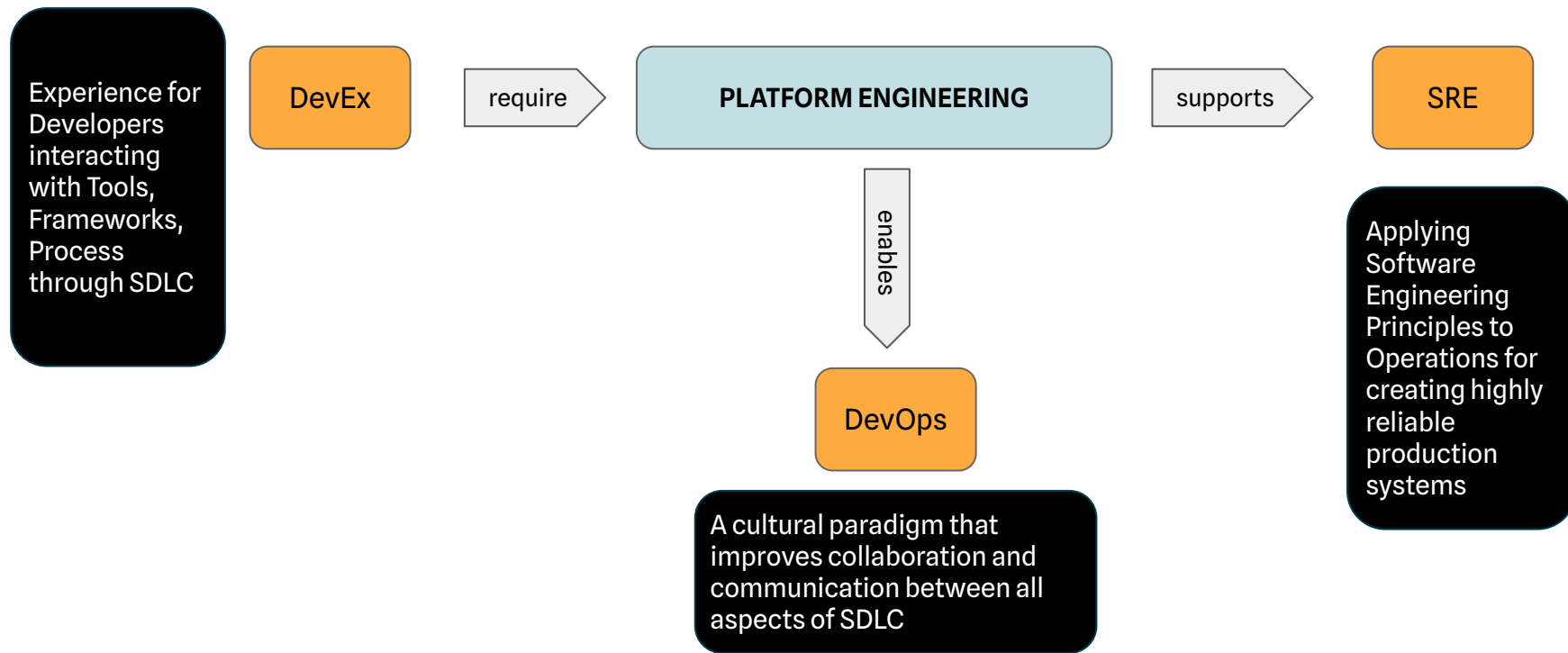


# Removing the friction with Platform Engineering





# Where does an engineering platform fit?



# Notional View of Platform Engineering

## Platform Product Management

Team Topologies, Technical Product Management,  
Value Modeling

### Developer Plane

Version Control, Infrastructure as Code , Dev Tools, Paved Road

### Compliance & Governance Plane

Pipelines , Lightweight governance, FinOps compliance, Compliance @ POC

### Delivery & Runtime Plane

Containers, Kubernetes, Workflow orchestration

### Networking & Connectivity Plane

VPC, External, 3rd party

### Security Plane

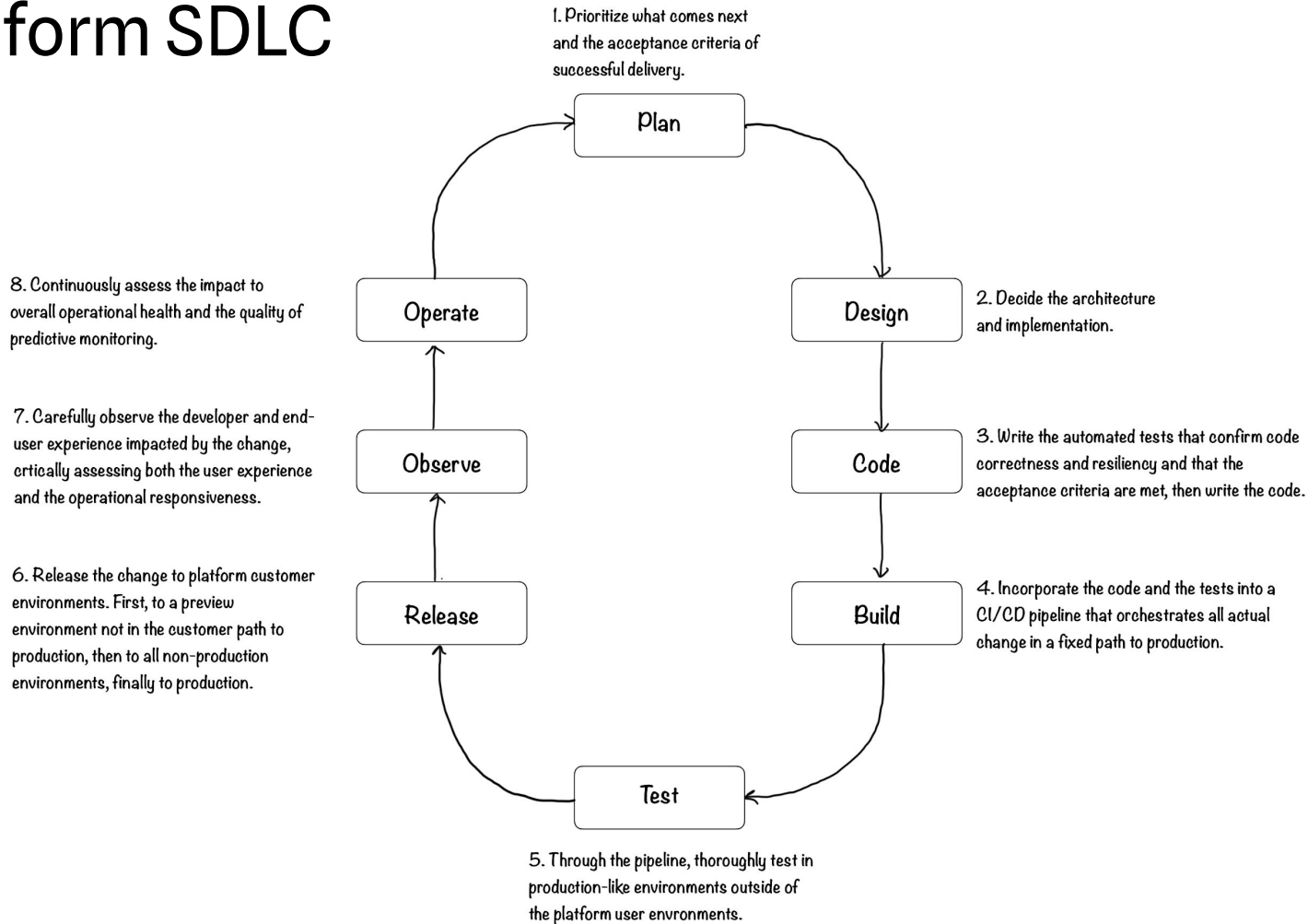
IAM, Secret and Encryption Management, SIEM

## Observability

System level, Integrations, Alerting

# Developing your platform as a composable software product

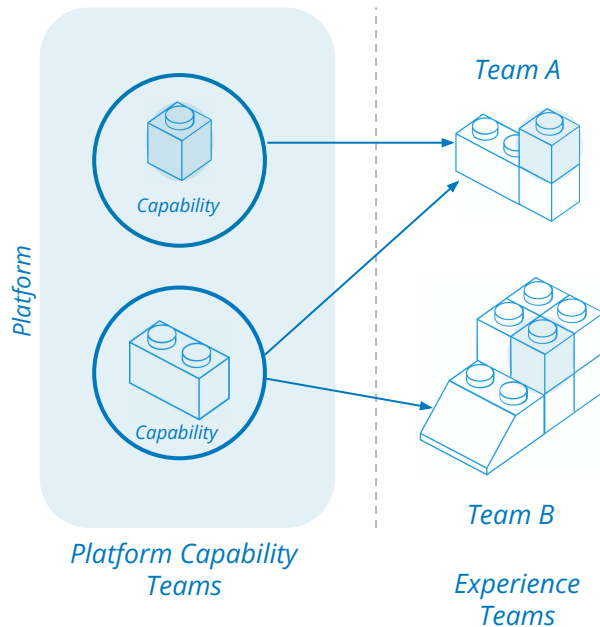
# The Platform SDLC



# Composable Capability Design

Platform capabilities are needed to leverage opportunities

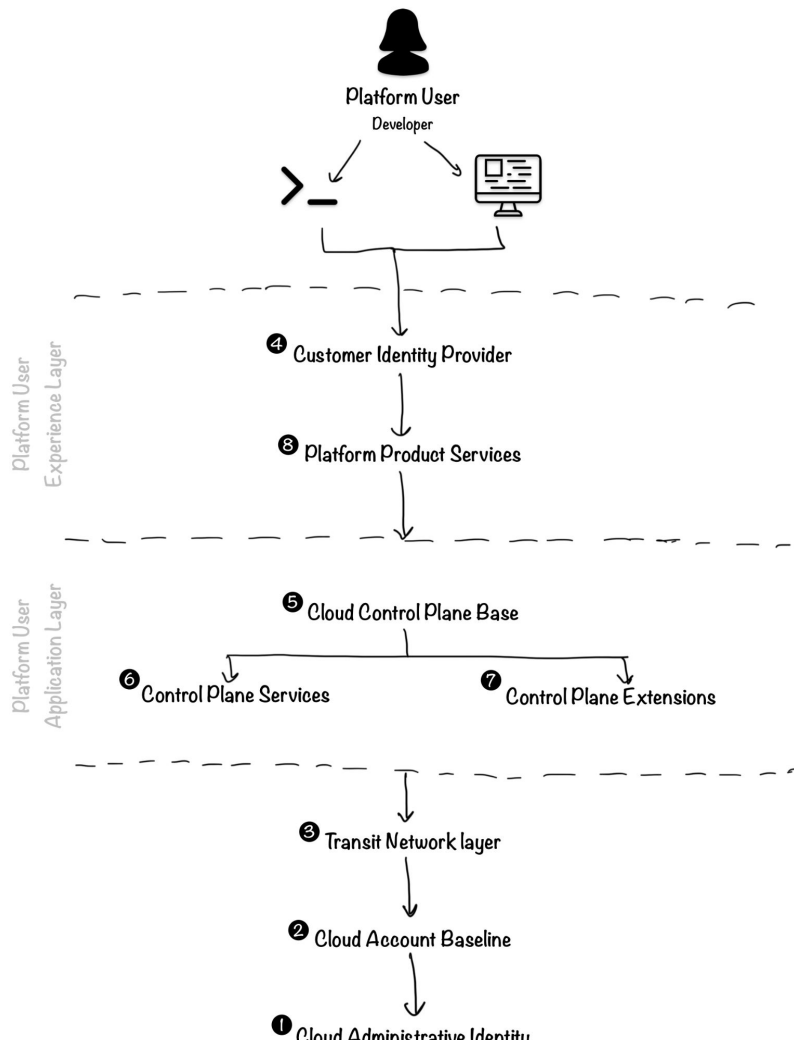
*Treating development teams as customers provides scale for innovation.*



# Platform Composable Design

Organizing your code by platform domain

*Organizing our code by platform domain sets up for scale from the beginning*



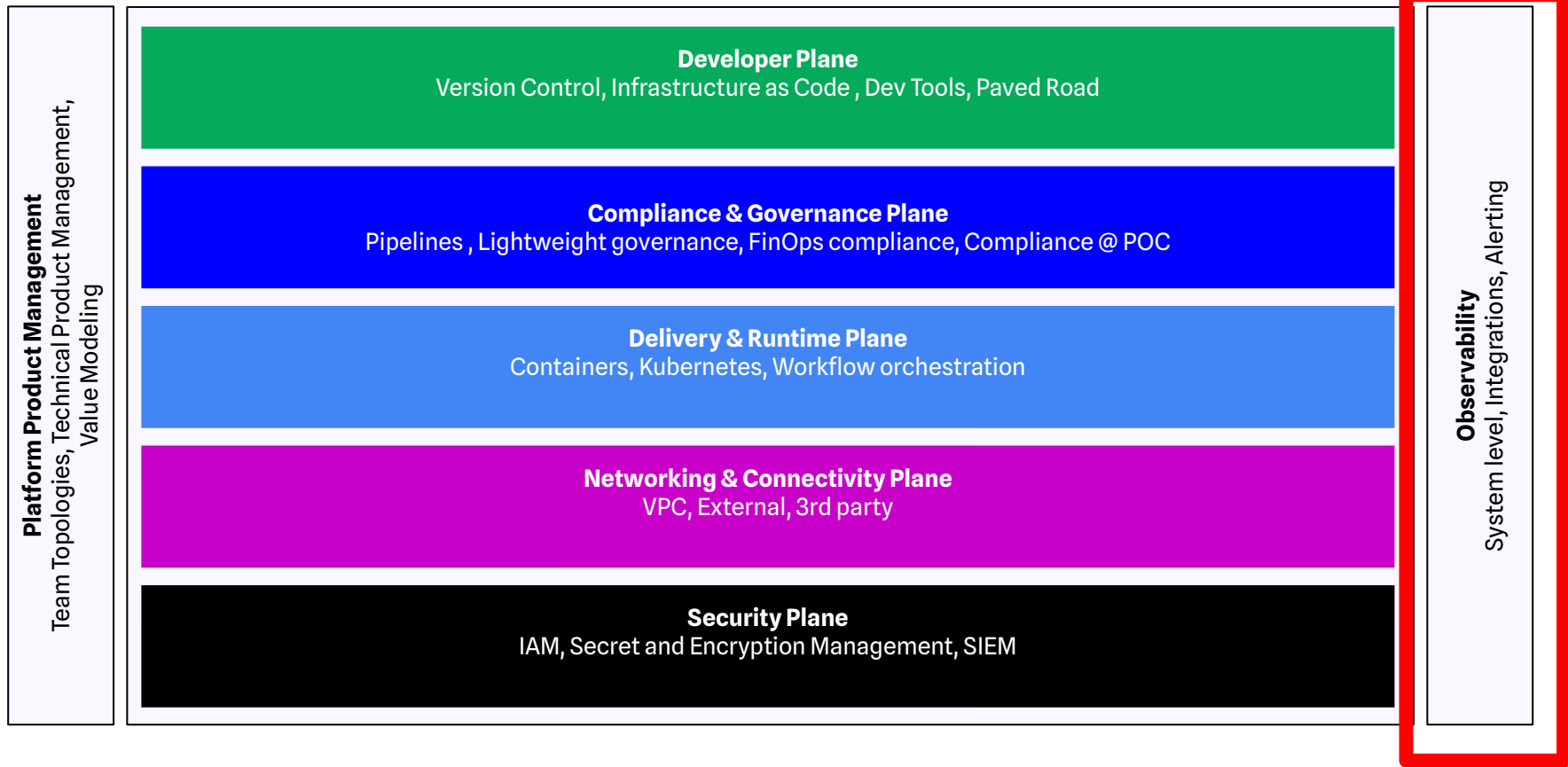
Let's look at a real example!

<https://bit.ly/3GYXdFz>

# Embedding observability for the platform and its users

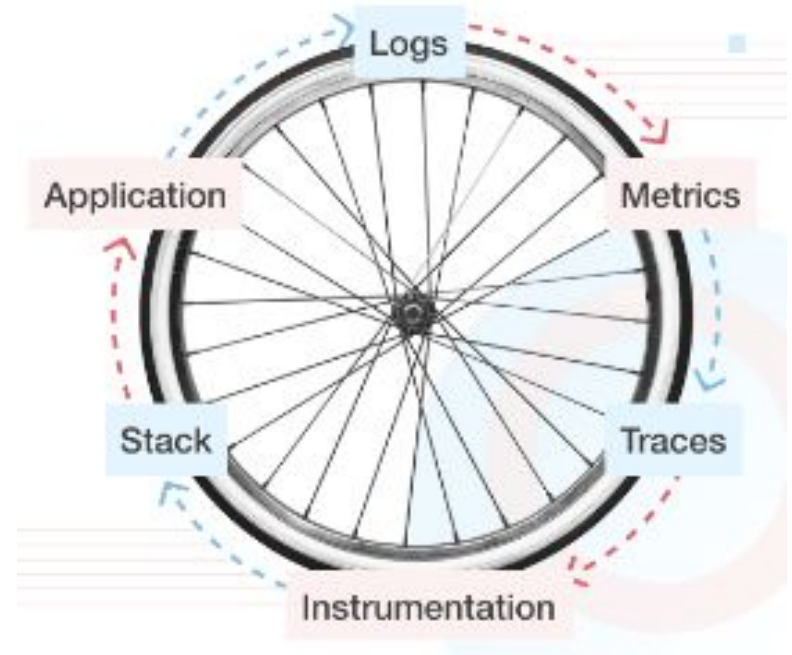


# Should Observability be an External Platform?



# Observability-Driven Development (ODD)

***Observability Driven Development (ODD)*** is a technique for building systems and software by shifting left observability concerns to the earliest parts of development.



# Embed Observability Throughout

## Platform Product Management

Team Topologies, Technical Product Management,  
Value Modeling

### Developer Plane

Version Control, Infrastructure as Code , Dev Tools, Paved Road

### Compliance & Governance Plane

Pipelines , Lightweight governance, FinOps compliance, Compliance @ POC

### Delivery & Runtime Plane

Containers, Kubernetes, Workflow orchestration

### Networking & Connectivity Plane

VPC, External, 3rd party

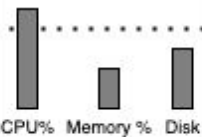



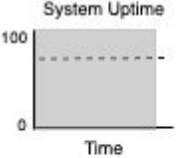

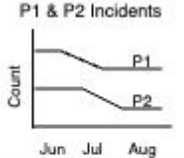

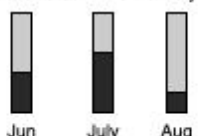

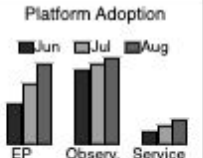





### Security Plane

IAM, Secret and Encryption Management, SIEM

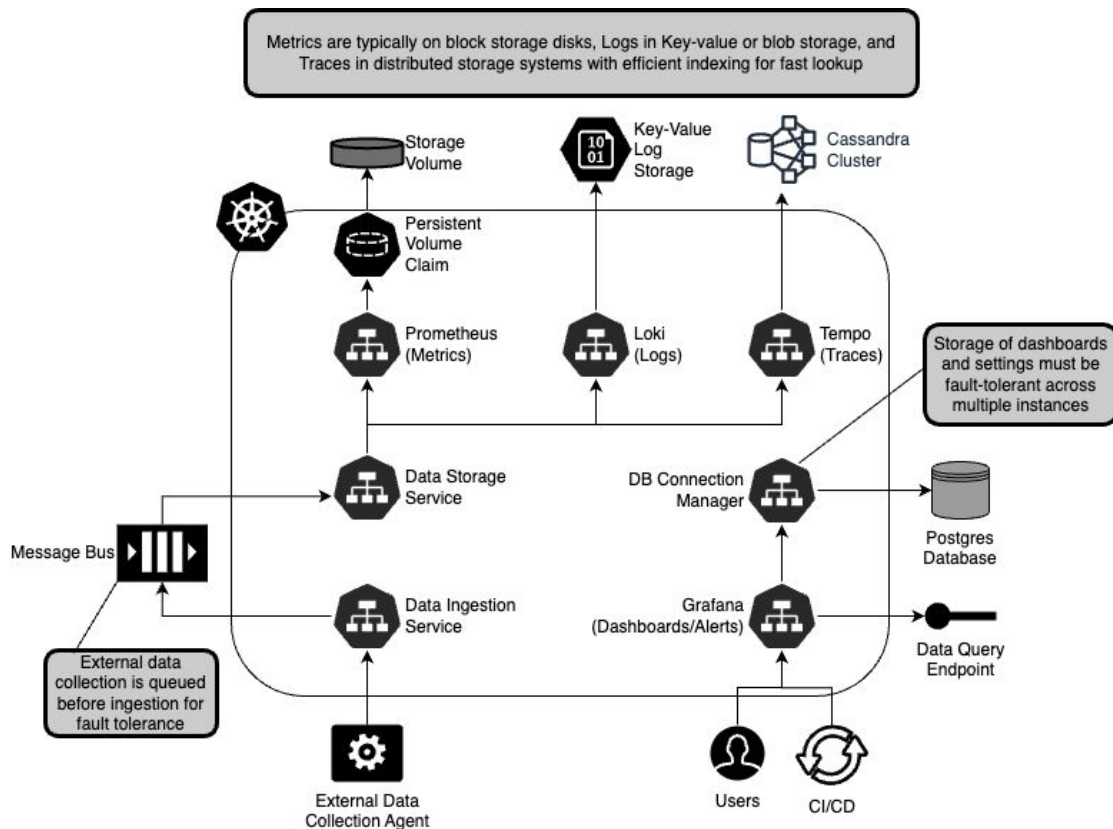
## Observability

System level, Integrations, Alerting

# What should we be observing?

<b>Engineering</b>	<p>1. Ensure usage and consumption are within limits</p> <div data-bbox="285 298 508 592"><p><b>Infrastructure</b></p><p>CPU% Memory % Disk</p><p> <b>Operations</b></p></div>	<p>2. Ensure applications are performing as expected and not throwing errors</p> <div data-bbox="653 298 875 592"><p><b>Application</b></p><p> <b>Engineering Teams</b></p></div>	<p>3. Ensure applications are available to serve requests</p> <div data-bbox="1020 298 1242 592"><p><b>Service Health</b></p><p> <b>SRE</b></p></div>	<p>4. Ensure issues are being addressed within acceptable times</p> <div data-bbox="1387 298 1609 592"><p><b>Incidents</b></p><p> <b>Incident Response</b></p></div>
<b>Business</b>	<p>5. Ensure delivery teams are performing as expected</p> <div data-bbox="285 723 508 1018"><p><b>Portfolio</b></p><p> <b>Portfolio Manager</b></p></div>	<p>6. Ensure that platform services are being used by teams and returning expected value</p> <div data-bbox="653 723 875 1018"><p><b>Platform</b></p><p> <b>Product Owners</b></p></div>	<p>7. Ensure that cloud services are being used responsibly by teams</p> <div data-bbox="1020 723 1242 1018"><p><b>Cloud</b></p><p> <b>Finance</b></p></div>	<p>8. Ensure that initiatives are leading to expected business results</p> <div data-bbox="1387 723 1609 1018"><p><b>Business Ops</b></p><p> <b>Executives</b></p></div>

# Architecture of an Observability Platform

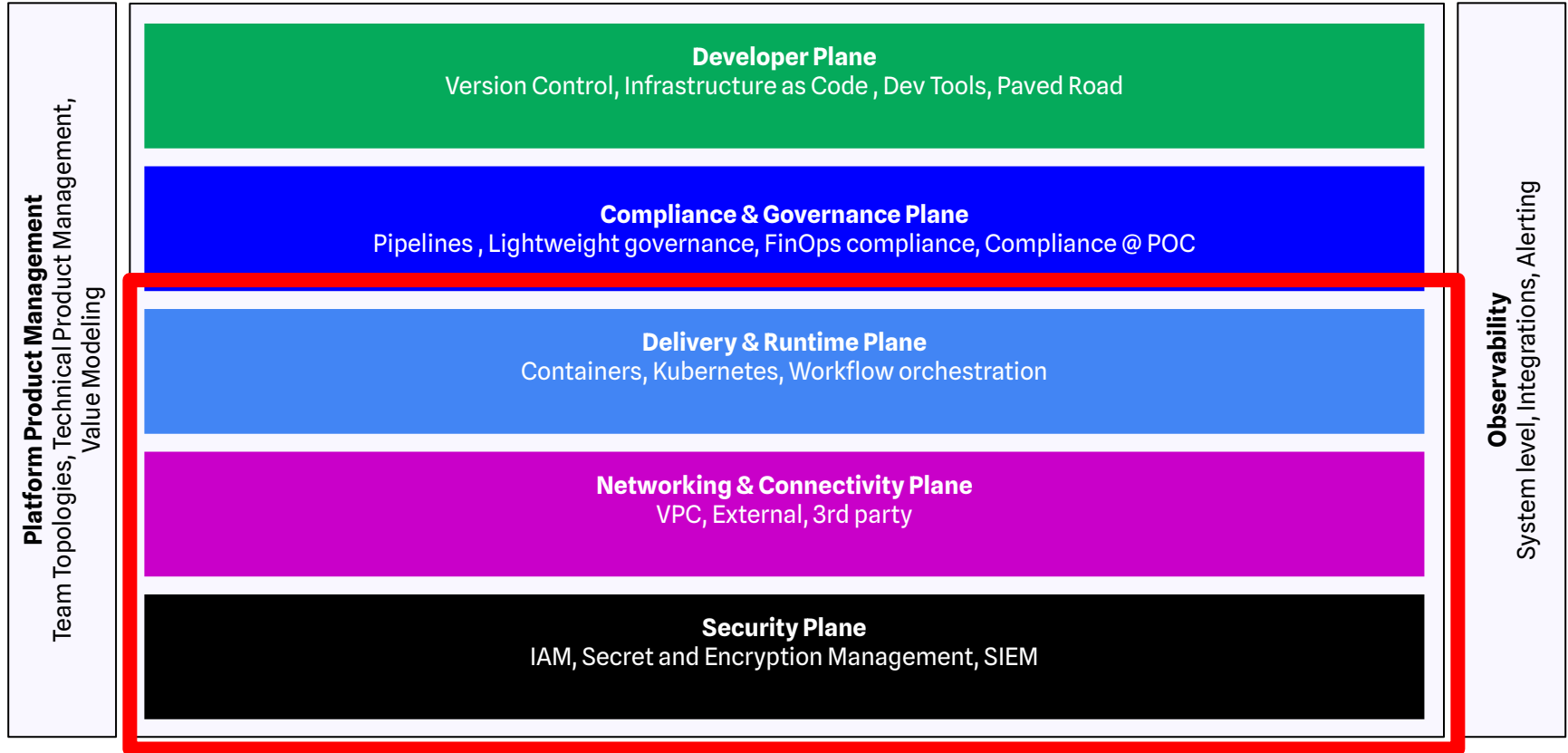


Let's see how we can  
deploy!

<http://bit.ly/40riFd2>

**Developing and deploying the control plane**

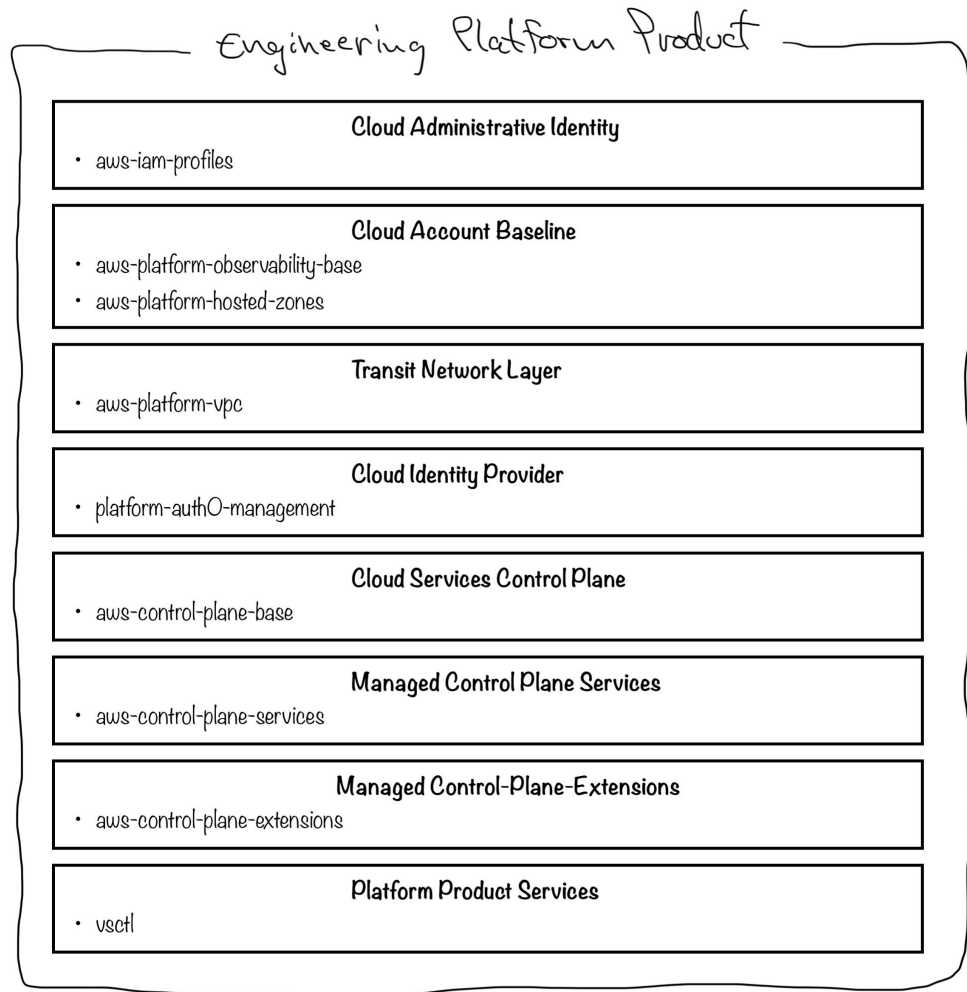
# Notional View of Platform Engineering



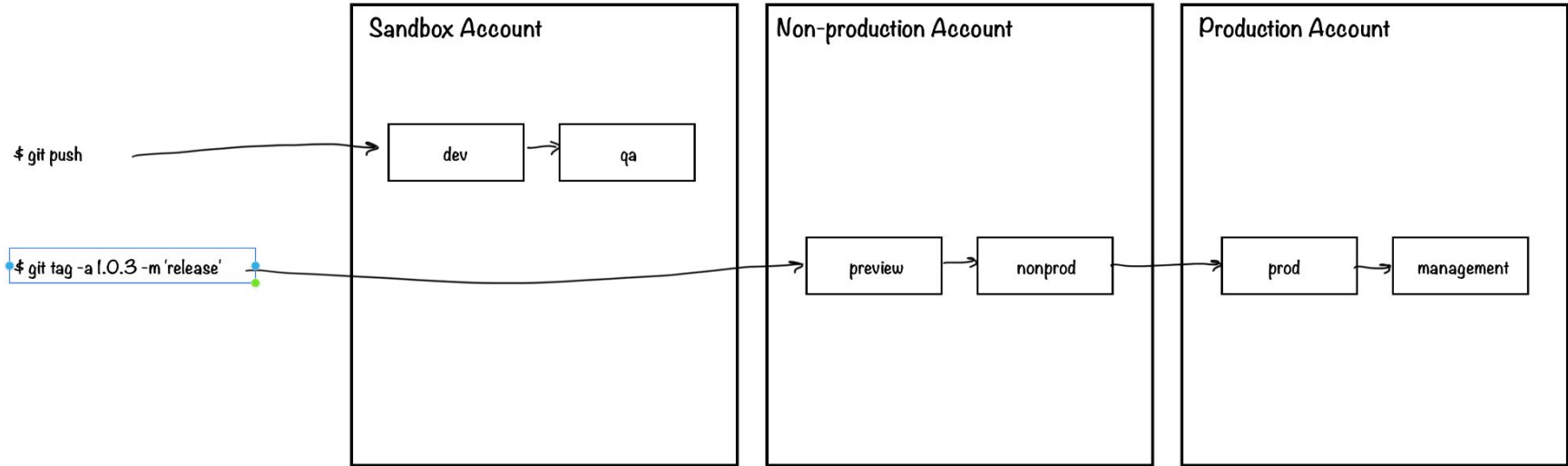


# Managing Control Plane Dependencies

*There may be dependencies early in development, but independent deployments can be enabled quickly*



# Environments We Need




# Let's look at the code!

<http://bit.ly/4dmFmEJ>

**Extending the platform to enable  
self-service for users**

# The Goal: Paved Paths

Not all applications are the same, but with consistent architectural patterns and deployment practices, a paved path can be provided to make getting code to production easy from day 1



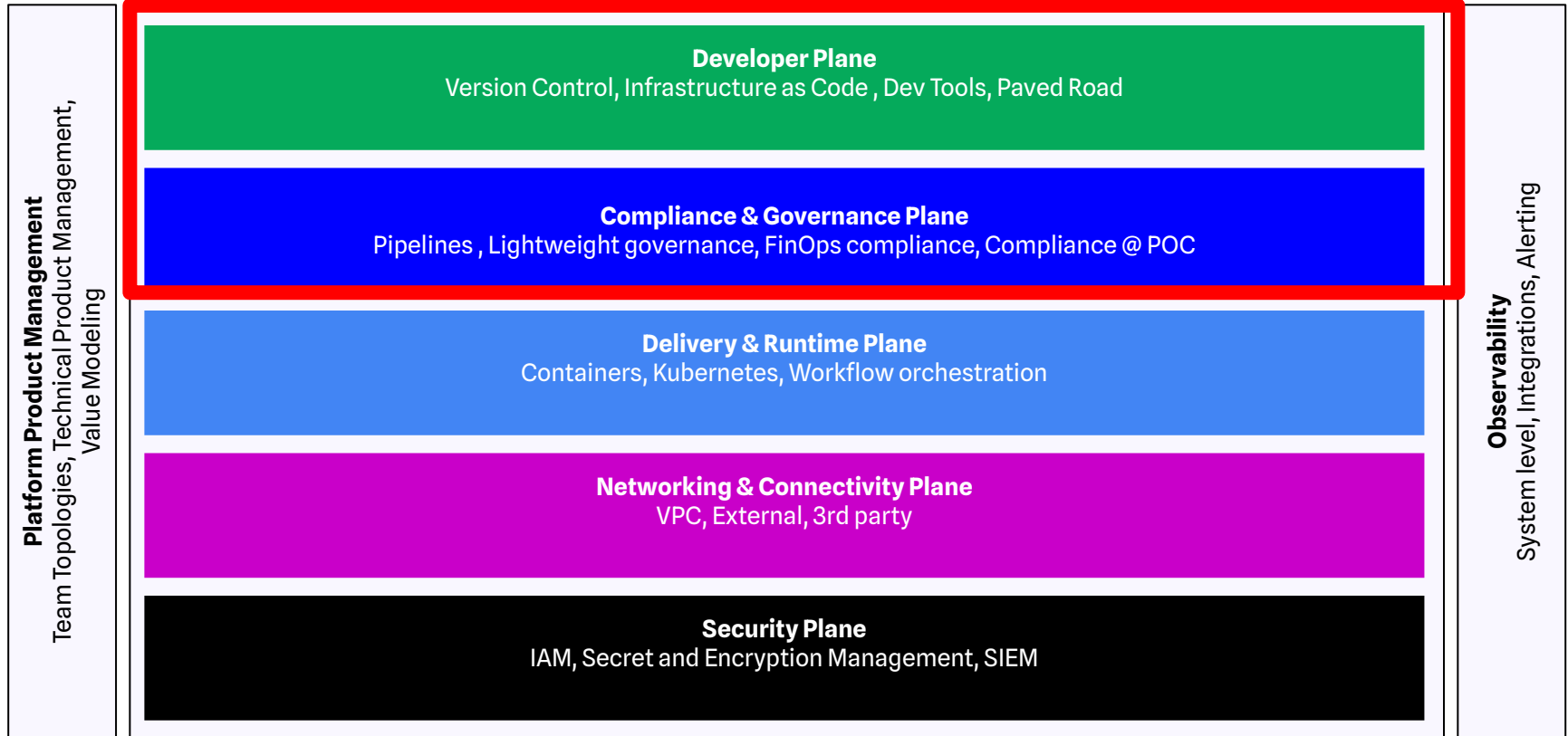
**Deploy to  
production right  
away**

**Ensure best  
practices and  
compliance**

**Extend and  
customize  
deployments**

**Diverge from the  
paved road in part  
or whole**

# Notional View of Platform Engineering



# What Extensions Are Needed?

As usual,  
it depends!!

## services

- observability collectors
- vertical autoscalers
- external service integrations
- policy and security scanning

Collect, analyze, and return or forward data about the state and activities of the cluster or applications running on the cluster.

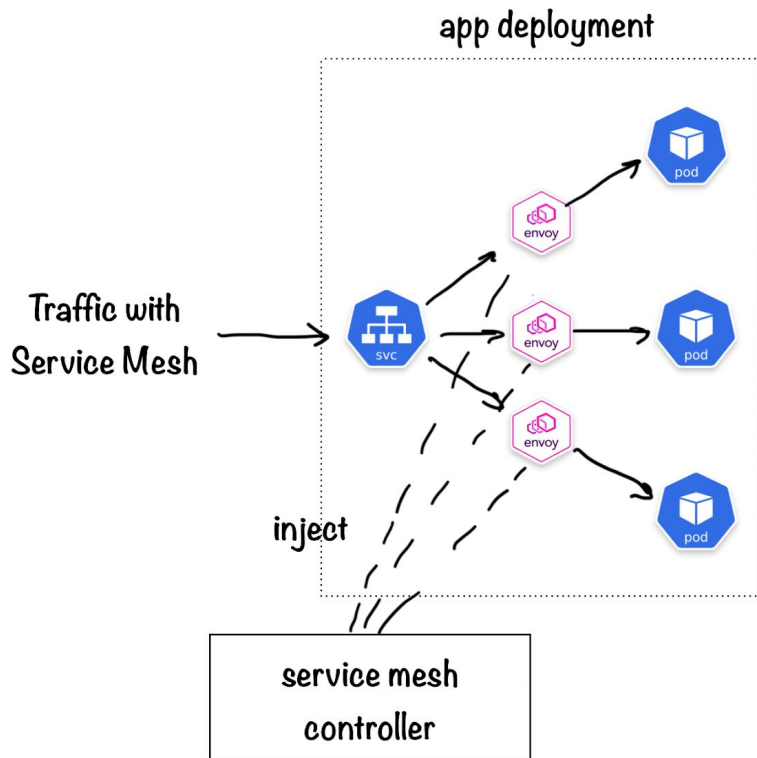
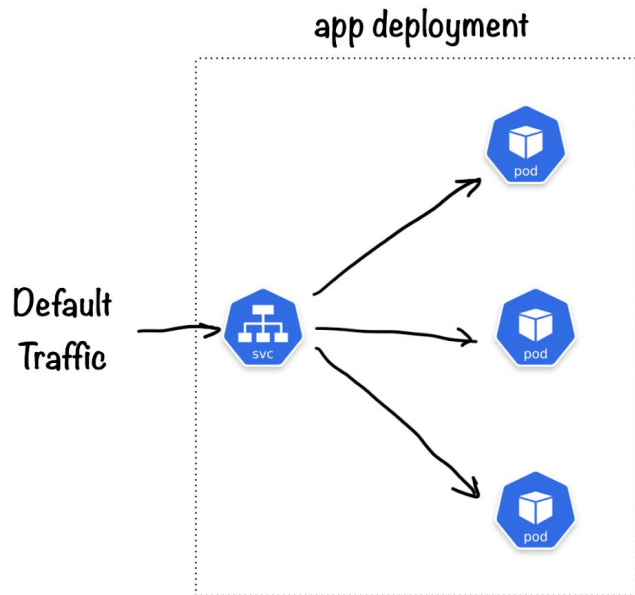
May provision or configure other resources inside or outside the cluster, but **ONLY** where such resources can be effectively fully abstracted from the platform user experience.

## extensions

- storage class
- infrastructure operator
- service mesh

Extend the Kubernetes API by enabling platform users to provision and configure a resource, whether inside or outside of the cluster, that is not part of the Kubernetes API definition.

# Enabling a Service Mesh



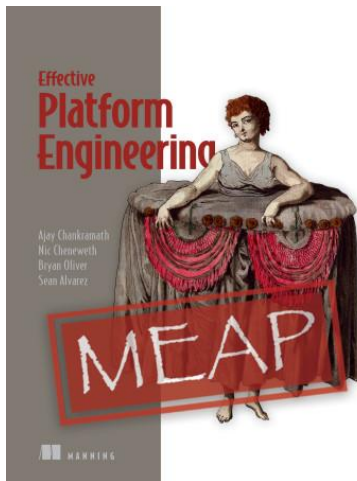


# Let's look at the code!

<https://bit.ly/4jXXSpb>

# Effective Platform Engineering

- **Effective Platform Engineering** book by [Manning](#)
- MEAP currently out awaiting print version soon




# Next Video - Impact of GenAI and LLMs in Platform Engineering

- Measuring improvements in platform engineering through AI methodologies
- How to improve developer experience through Generative AI
- How is Generative AI improving the observability space?
- What is Agentic AI and how does it impact platform engineering?



# More Information



<https://effectiveplatformengineering.com/>



## Effective Platform Engineering





[Buy Now](#)

  
[Ajay Chankramath](#)  


  
[Nic Cheneweth](#)  


  
[Bryan Oliver](#)  


  
[Sean Alvarez](#)  


"Effective Platform Engineering" is a comprehensive guide that introduces platform engineering as a discipline, focusing on creating developer platforms that enhance team efficiency and streamline application deployment. The book provides practical insights into designing and managing platforms that bridge the gap between operations and development, automating tasks throughout the software development lifecycle. Readers will learn to build internal developer platforms and portals, ensuring seamless adoption and satisfaction among teams. The authors emphasize the importance of secure, scalable Kubernetes-based engineering platforms and offer strategies for implementing effective Service Level Objectives to boost trust and adoption. Additionally, the book explores cutting-edge integrations of Generative AI tools to enhance developer productivity, providing readers with the knowledge to leverage the latest advancements in code generation.

Through practical examples and real-world scenarios, "Effective Platform Engineering" demonstrates how platform engineering differs from traditional DevOps and the unique value it brings to organizations. The book delves into both patterns and anti-patterns of platform development, guiding readers in designing and deploying secure, scalable, and observable engineering platforms. With the inclusion of diagrams, code samples, and exercises, readers can visualize key concepts and solidify their understanding. This resource is tailored for DevOps engineers familiar with Kubernetes, cloud environments, and infrastructure-as-code, aiming to equip them with the skills to establish platforms that reduce workloads, improve consistency, and accelerate software delivery.

Discover how platform engineering is revolutionizing the developer experience and operational efficiency. [Learn more](#)